

Re: Blue Chip Technology + MagnumX?

Source: <http://coding.derkeiler.com/Archive/General/comp.arch.embedded/2006-08/msg01213.html>

- *From:* David Hearn <dave@xxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Wed, 16 Aug 2006 11:09:19 +0100
-

Paul Carpenter wrote:

On Tuesday, in article <4kdgojFbg7edU1@xxxxxxxxxxxxxxxx>
dave@xxxxxxxxxxxxxxxxxxxxxxxx "David Hearn" wrote:

John Devereux wrote:

David Hearn <dave@xxxxxxxxxxxxxxxxxxxxxxxx> writes:

John Devereux wrote:

David Hearn
<dave@xxxxxxxxxxxxxxxxxxxxxxxx>
writes:

Has anyone
got any
experience
of Blue
Chip
Technology
(<http://www.bluechiptechnology.co.uk/>)
and
especially
their
MagnumX
single board
computers?
(http://www.bluechiptechnology.co.uk/single_board_computer.php?s)

=2)

We've been
investigating
the
ColdFire
MCF5475EVB
for some

Re: Blue Chip Technology + MagnumX?

quite simple
5 channel
GPIO
capture –
but at quite
high data
rates. We've
got the
266MHz
ColdFire
processor to
do what we
want but it
just isn't
quite
fast enough
– it's
missing
some of the
state
transitions
on the
channels.
We're
sampling at
around 2 to
2.2MHz and
we need
more like
3.3MHz.

We were
wondering
whether a
faster (2x?)
processor
would be
more
likely to do
the job. The
733MHz
MagnumX
board
sounds like
it might
do this, and
it has 16
channels of
GPIO (we
only need 5
inputs at

Re: Blue Chip Technology + MagnumX?

present,
unlikely to
increase
significantly).
I'm aware
it's moving
from the
ColdFire/M68K
processor to
an x86
processor,
but unless
there's a
major
difference
in per cycle
processing,
that's not a
problem.

Our app is
quite simple
– loop until
a counter
expires,
each
iteration
store 1 byte
pin state
register and
32 bit
counter
value. We
currently
can sample
up to 60MB
of data.

Can you not use DMA
instead? And/Or perhaps an
external FIFO.

Personally, I have no idea. I've never used
DMA before (I'm
traditionally a desktop app guy!) – so I'm not
sure where I would
start, or what I'd need to do. Essentially, we
just want to move a 8
bit GPIO register containing pin status and a
32 bit register
containing a counter value into RAM as fast

Re: Blue Chip Technology + MagnumX?

as possible – at around
3.3MHz (sample every 300ns).

Paul Carpenter beat me to most of the points I would make. I would just wonder whether you actually need the counter – if the samples are being acquired reliably then perhaps the sample number can give you the time?

Yes, if the samples are being taken at guaranteed periodic intervals then use of the counter is not needed.

The software counter, only *guarantees* that a counter variable counts it can not under all circumstances guarantee anything else.

The counter used decrements once per (133MHz) clock tick. Difference between counter values can then be used to determine period between samples.

Originally to save RAM (we 'only' have 64MB on board), we only logged the pin state if it had changed, but this added a little processing into the loop (thus making it slower), and also required the use of the counter to get timing values as the time for each iteration would change depending on whether the pin state had changed. When the pin state was the same the loop was shorter, when it had changed it was longer.

Does the time factor between samples get logged as well as the changes?
If not the data starts to become meaningless.

Yes, when we did filtering of samples during the capture, the counter (ie. the time factor) was used to determine the period between samples. When we removed the filtering, and had a consistent code path for each sample iteration, the need for storing the interval was gone.

What is the maximum frequency of change of ANY digital input compared to desired sampling frequency?

There are 5 digital inputs. Currently 4 of them are sampled perfectly fine and provide the data we're after. The issue is with 1 input which can run around 850kHz if data is present. It's not a stable line (ie. not a 850kHz clock) but has frequent phase changes – but the max is around 850kHz.

At present, the 2.xMHz we're sampling at isn't quite fast enough to get all the transitions. On the other 4 lines the speed is fine.

Re: Blue Chip Technology + MagnumX?

Re: Blue Chip Technology + MagnumX?

To try and speed things up (ie. reduce memory accesses and conditionals), we started just sampling the pin state, but not filtering out any non-changes – therefore making the code path in each iteration of the loop identical. This, in theory, should give the same sample rate each time, once the 'fudge' factor (time taken for each loop iteration) is calculated.

If you can guarantee that NO other actions could EVER occur on your board, causing the micro or its support hardware to put in waits and change your timing.

So far we've not seen any signs that this is happening, or at least, if it's happening, it's not affecting the quality of our (4 line) samples.

I believe this worked well – however the sample rate was still only fractionally faster than 2MHz – not close to the (approx) 3.3MHz we'd like.

The average sample rate, over a period of time, we have no idea what accuracy of capture timing you are trying to achieve.

At the rates of capture you are trying to achieve I would NEVER do it by software polling.

Using the same code in an interrupt handler (store byte in array, increment index) was taking longer than polling.

Anyway, DMA uses special hardware facilities to move data around, rather than doing it in software. Look it up in the datasheet for your coldfire chip. You should be able use the output of a timer to trigger a DMA transfer from a PIO port to memory, at a programmed rate.

I'll look into that. The issue I found yesterday whilst looking at some generic DMA stuff is that the amount of code to initiate a DMA transfer is significant, much more than what is currently occurring in the loop. Whilst the memory transfer might be faster, the actual processing to initiate the transfer will be greater.

If you want to capture many megabytes then most DMA controllers are capable of dealing with 32K and larger transfers fully under hardware timed control (if DMA is timer triggered).

Re: Blue Chip Technology + MagnumX?

I suggest you do some reading up on DMA on ANY processor and the concepts will become clear.

Will do.

I guess this is what Paul meant by using a deep FIFO (I'm still learning!). This can be used to buffer up the samples such that the DMA transfer can be done on a larger block, rather than the 1 (or 5) bytes I'm currently processing each time.

The large FIFO means that you know the time interval between EACH sample point because external hardware times the writes to the FIFO. The reads from the FIFO are subject to system software, hardware latencies and on most DMA controllers the time interval to setup the controller to point to the next memory block to use. The deeper the FIFO allows for longer latencies (accumulative) and longer delays (next memory block setup time). Also cover for sloppy code and other system events.

Am I correct in saying that using DMA to transfer 1 or 5 bytes (if including a counter) at a time is likely to have more overhead than just doing a normal array indexed write?

Comparing transferring a few bytes from memory to memory, or a SINGLE transfer from I/O to/from memory, then YES.

Comparing controlled timing to LARGE amounts of data to/from I/O at timed intervals, timer control (internal or external) and DMA wins hands down over longer periods.

I wasn't aware (until John pointed out) that DMA was a setup and forget system – and the overhead is in the setup, not the subsequent transfers. I can see now that for longer periods, DMA should be better.

Judicious use of multiple DMA buffers, would also allow your software to be 'compressing' the data from ONE buffer to change events to save, AT THE SAME TIME as `_hardware_` is filling up another buffer, The amount of buffers and their size depends on the

- System in use
- Length of data run
- Time to process one buffer (MAX time) by software.
- Capture rate (MIN time to fill a buffer)
- Other system activity
- Other things I have forgotten.

Re: Blue Chip Technology + MagnumX?

We originally tried using a timer to generate interrupts and to sample at a constant rate, however, we found that the sampling speed was far slower than using a simple while() loop – too slow for our needs. To

Probably too much code in the interrupt routine and interrupt latencies.
The timer should be DIRECTLY strobing the data capture.

See above about code in interrupt handler and performance issues.

further complicate matters, turning full optimisation on (using gcc for M68k) caused interrupts to break. Using full optimisation made a noticeable difference in speed when doing the simple while() loop.

Sounds more likely to be improper use of things like volatile and other software design problems.

If -O3 is enabled, then when entering an interrupt handler and then clearing/acknowledging the interrupt, when exiting the same interrupt got fired again – implying that the clearing of the interrupt never occurred. Turning it down to -O1 (and no other changes) again made it work. Register .h file definitions are from Freescale/LogicPD provided SDK/header files.

Unfortunately I have little control over modifications of the hardware (ie. adding FIFOs etc). Whilst I can advise what might be a better off the shelf choice, custom changes to boards is unlikely to happen at this stage (proof of concept and initial development).

Well if you don't sort out the data acquisition properly BEFORE deciding on processor, it is akin to saying

"I have this wonderful MP3 player which needs an Edison Wax Cylinder interface via USB without designing the hardware to interface it."

Proof of concepts have always in my opinion have to show what is and what is NOT needed. If you approach the problem with the wrong tool then you will need to add something to kludge the lot together.

All you are proving at the moment is that we can throw money at buying bits and any bits, that if we keep buying bits we will force the problem to be solved.

There are two options we are considering to solve the problem.

- 1.) Sample faster (from 2.xMHz to 3.3MHz)

Re: Blue Chip Technology + MagnumX?

Re: Blue Chip Technology + MagnumX?

2.) Get the designer of the board generating the digital inputs to detect the phase changes and have that as the 5th digital input, rather than the 850kHz data we're getting. This would mean our current implementation will be suitable.

Work out what needs to be done how and under what constraints first, then find the necessary hardware and software. You APPEAR to have the hardware and software which you are trying to force into meeting unknown constraints.

.....

I'm certainly no true embedded engineer – I'm traditionally a PC software engineer who hadn't had to worry about low level code before. In the (small) company, I probably have the most 'embedded' experience through doing some set top box software development (using provided SDKs not requiring low level hardware interfacing). This has been my first true embedded development project, and had a steep learning curve but has been fun. My knowledge of architectures and embedded hardware is very limited. The Coldfire board was selected by someone else (who's moved on now!) and if I can find a more suitable architecture/board then that would be good. I'm not in a position to design any hardware really – just select something that's already out there.

Well you need to find some form of CLOCKED digital input to some form of DMA transfer to some processor. They exist but mainly for PCs and the like, not many of them. Long time since I looked for such things. They are also normally relatively expensive items.

I have no experience of selecting, programming or interfacing with FPGAs, but I have had it suggested before that something like this might be better suited to this task. The problem is, I'm not a hardware design guy.

It seems to me that the system designer or whoever is in charge of the overall system design needs to review the 'proof of concept' strategy.

That would be me then.

I've even had a suggestion along the lines of "just use a memory controller chip and use a clock to transfer the pin status into the RAM directly" (or something like that). Sounds a great idea – but that's not something I can do with my current skills.

Someone where you work must be able to do it.

Re: Blue Chip Technology + MagnumX?

Team of 5, I'm the only one with *any* hardware experience, and that was over 6 years ago during my degree during which I biased towards the software side over the (mostly analogue and RF) hardware.

I appreciate all your help – but really, at the moment, we're limited to using an off-the-shelf processor board – it's just knowing which is suitable. We're 2/3 of the speed we need at present – so not a huge amount to make up – it feels as if we should be able to do it using our current 'design' – just wondering whether other hardware would have the speed increase we're looking for.

Other hardware added to your board or even much lesser processor cards will easily do this FUNCTION. Other hardware will do something your hardware will not do, guarantee the sample to sample timing ensuring that it stays that way is moving the data as fast as possible into memory to keep up with sampling regime.

I don't disagree with this statement.

You appear not to have done any capture timing tests to see how regular your digital capture is from sample to sample and whether it meets any part of your spec.

We have done this, and confirmed that 4 of the 5 digital inputs are being sampled with sufficient quality to consistently provide the data we're looking for.

All you have at the moment shown to me is that you have a bunch of data acquired at AVERAGE clock rates.

Which is fine for our purpose (except the average rate is too slow for one of the inputs, which we may be able to solve with a change elsewhere).

Having spent many years dealing with all sorts of data acquisition and processing upto and including high speed Video data and down to slow heat measurements (and slower ones). This seems to me to be data acquisition of data with no reference to make the data mean anything useful.

I suggest that you get someone involved who can see the WHOLE system issue and is ABLE to if necessary design the correct hardware to support the functions required.

Much as I could be able to do such a thing I have several projects currently on the go from network infrastructure to ASIC testing. I am sure there

Re: Blue Chip Technology + MagnumX?

are plenty of people out there or in your organisation that could do this.

Incidentally, I got a reply from Blue Chip Technology regarding my question to them. They were very helpful and said:

"Unfortunately, the MagnumX will not do what you are looking for. On the board the GPIO lines run on the I2C bus, and the chipset manufacturers recommendation is that operation of these lines should be around 5Hz max, well below what you are looking for.

Looking at your requirements, our design engineers have suggested that you look at some form of Bus Mastering Data Acquisition card to get the speed you are looking for. Even then, with the 33Mhz PCI bus, and a PCI Read taking about 10 clock cycles per register, you are right on the limits of what the PCI bus can cope with.

I'm sorry but we cannot even think of any Data acquisition card that may be suitable"

I suspect a typo in the 5Hz max – but I believe the meaning is still the same.

Looks like a change to the input board may be the best option (if it's possible) as then we'd met our requirements and got the necessary data sampled to the resolution and quality we require.

Thanks again

D

.